In [1]:
```
import pandas as pd
movie=pd.read_csv(r'Downloads\Movie-Rating.csv')
```

In [2]:
```
import numpy as np
```

In [3]:
```
print(pd.__version__)
```

```
2.2.2
```

In [5]:
```
movie
```

Out[5]:

	Film	Genre	Rotten Tomatoes Ratings %	Audience Ratings %	Budget (million $)	Year of release
0	(500) Days of Summer	Comedy	87	81	8	2009
1	10,000 B.C.	Adventure	9	44	105	2008
2	12 Rounds	Action	30	52	20	2009
3	127 Hours	Adventure	93	84	18	2010
4	17 Again	Comedy	55	70	20	2009
...	...	...	...	...	...	...
554	Your Highness	Comedy	26	36	50	2011
555	Youth in Revolt	Comedy	68	52	18	2009
556	Zodiac	Thriller	89	73	65	2007
557	Zombieland	Action	90	87	24	2009
558	Zookeeper	Comedy	14	42	80	2011

559 rows × 6 columns

In [6]:
```
movie.columns
```

Out[6]:
```
Index(['Film', 'Genre', 'Rotten Tomatoes Ratings %', 'Audience Ratings %',
       'Budget (million $)', 'Year of release'],
      dtype='object')
```

In [7]:
```
movie.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 559 entries, 0 to 558
Data columns (total 6 columns):
 #   Column                     Non-Null Count  Dtype 
---  ------                     --------------  ----- 
 0   Film                       559 non-null    object
 1   Genre                      559 non-null    object
 2   Rotten Tomatoes Ratings %  559 non-null    int64 
 3   Audience Ratings %         559 non-null    int64 
 4   Budget (million $)         559 non-null    int64 
 5   Year of release            559 non-null    int64 
dtypes: int64(4), object(2)
memory usage: 26.3+ KB
```

In [8]:
```
movie.head()
```

Out[8]:

	Film	Genre	Rotten Tomatoes Ratings %	Audience Ratings %	Budget (million $)	Year of release
0	(500) Days of Summer	Comedy	87	81	8	2009
1	10,000 B.C.	Adventure	9	44	105	2008
2	12 Rounds	Action	30	52	20	2009
3	127 Hours	Adventure	93	84	18	2010
4	17 Again	Comedy	55	70	20	2009

In [9]:
```
movie.columns=['films','gener','rating','audiencerating','budgetmillions','year'
```

In [10]:
```
movie.shape
```

Out[10]:
```
(559, 6)
```

In [11]:
```
movie.columns
```

Out[11]:
```
Index(['films', 'gener', 'rating', 'audiencerating', 'budgetmillions', 'year'],
dtype='object')
```

In [12]:
```
movie.films=movie.films.astype('category')
movie.year=movie.year.astype('category')
movie.gener=movie.gener.astype('category')
```

In [14]:
```
movie.describe()
```

Out[14]:

	rating	audiencerating	budgetmillions
count	559.000000	559.000000	559.000000
mean	47.309481	58.744186	50.236136
std	26.413091	16.826887	48.731817
min	0.000000	0.000000	0.000000
25%	25.000000	47.000000	20.000000
50%	46.000000	58.000000	35.000000
75%	70.000000	72.000000	65.000000
max	97.000000	96.000000	300.000000

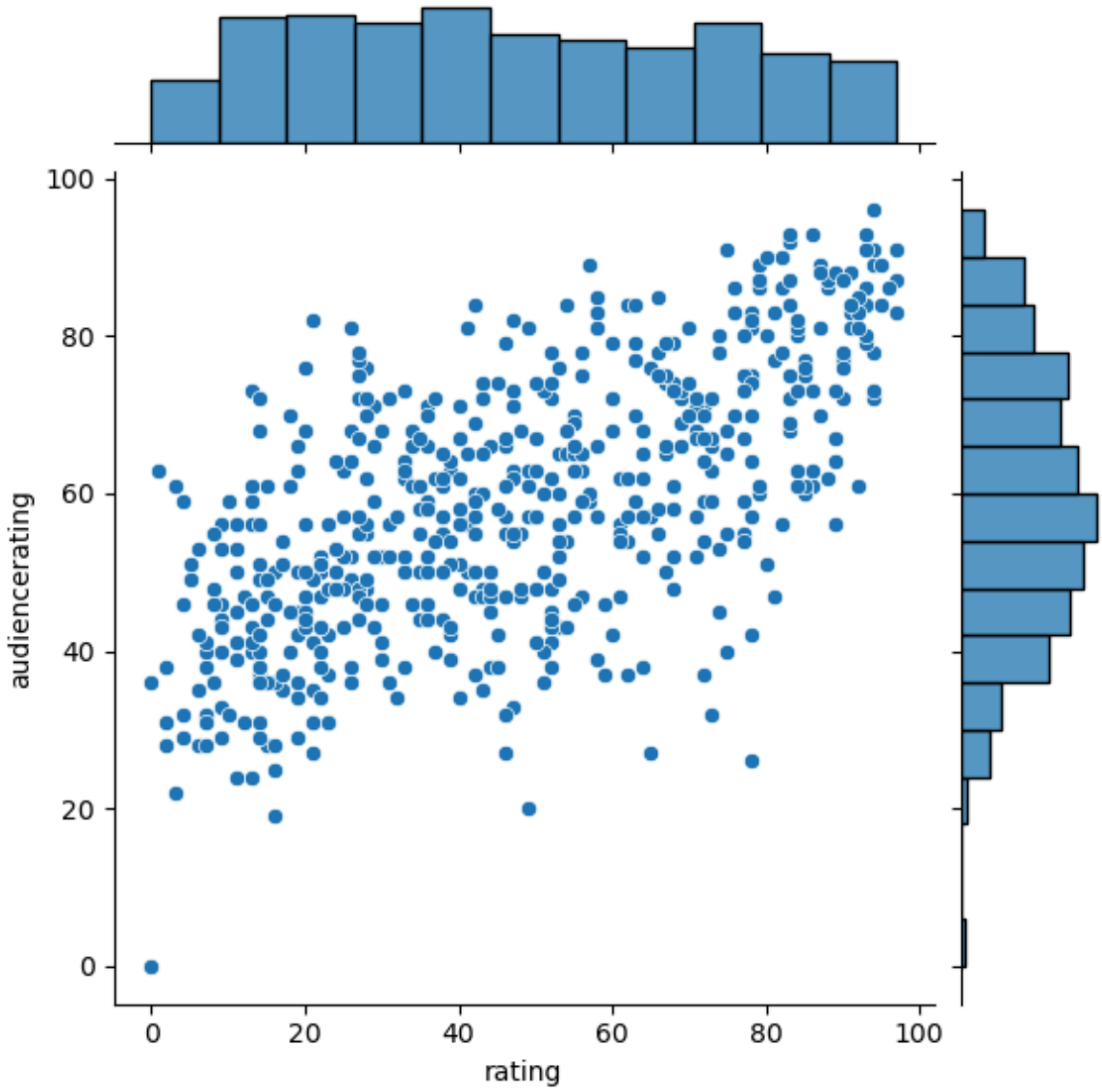
In [15]:
```
movie.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 559 entries, 0 to 558
Data columns (total 6 columns):
 #   Column          Non-Null Count  Dtype   
---  ------          --------------  -----   
 0   films           559 non-null    category
 1   gener           559 non-null    category
 2   rating          559 non-null    int64   
 3   audiencerating  559 non-null    int64   
 4   budgetmillions  559 non-null    int64   
 5   year            559 non-null    category
dtypes: category(3), int64(3)
memory usage: 36.5 KB
```

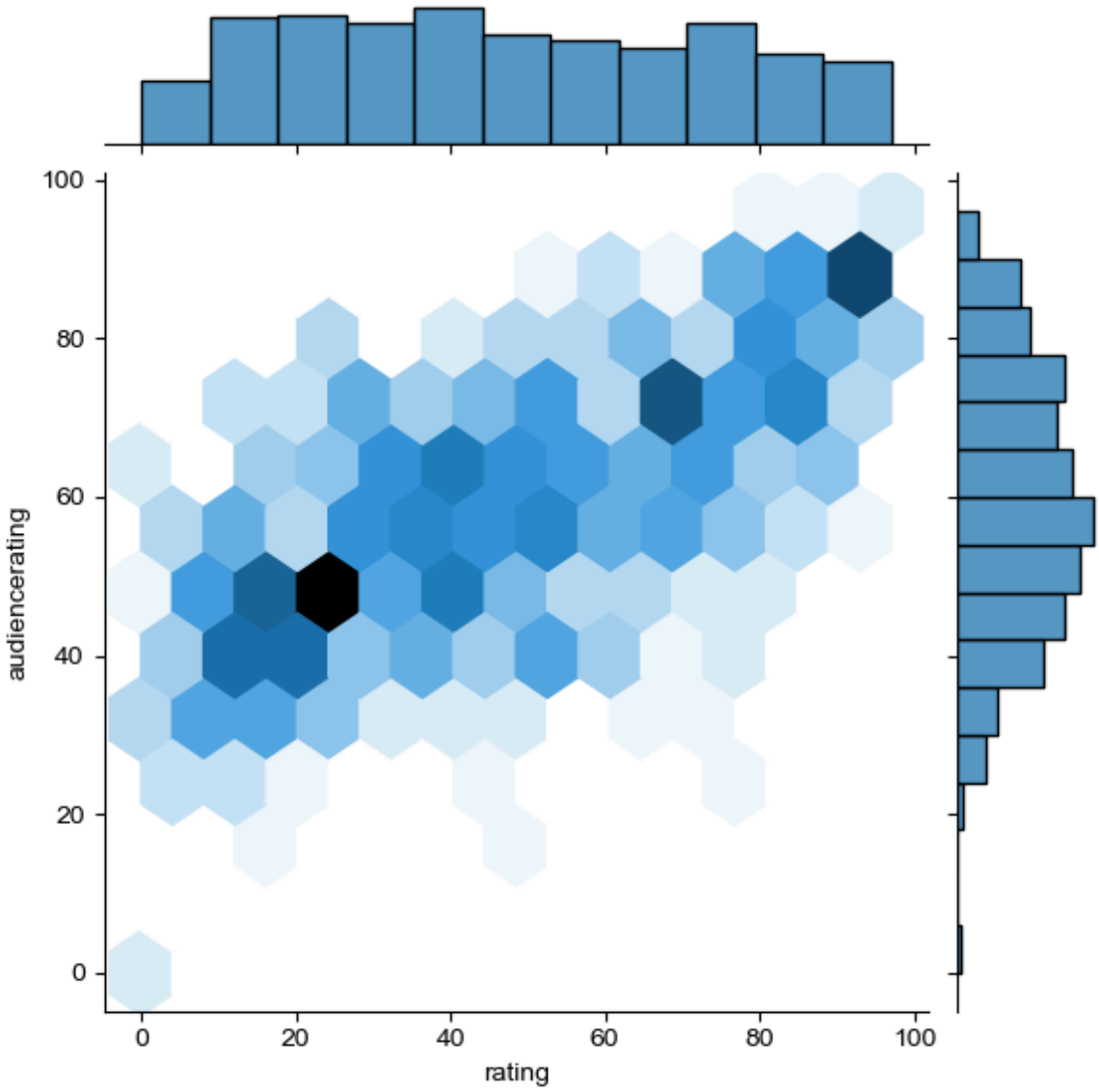
In [16]:
```
from matplotlib import pyplot as plt
import seaborn as sns
%matplotlib inline
import warnings
warnings.filterwarnings
```

Out[16]:
```
<function warnings.filterwarnings(action, message='', category=<class 'Warnin
g'>, module='', lineno=0, append=False)>
```

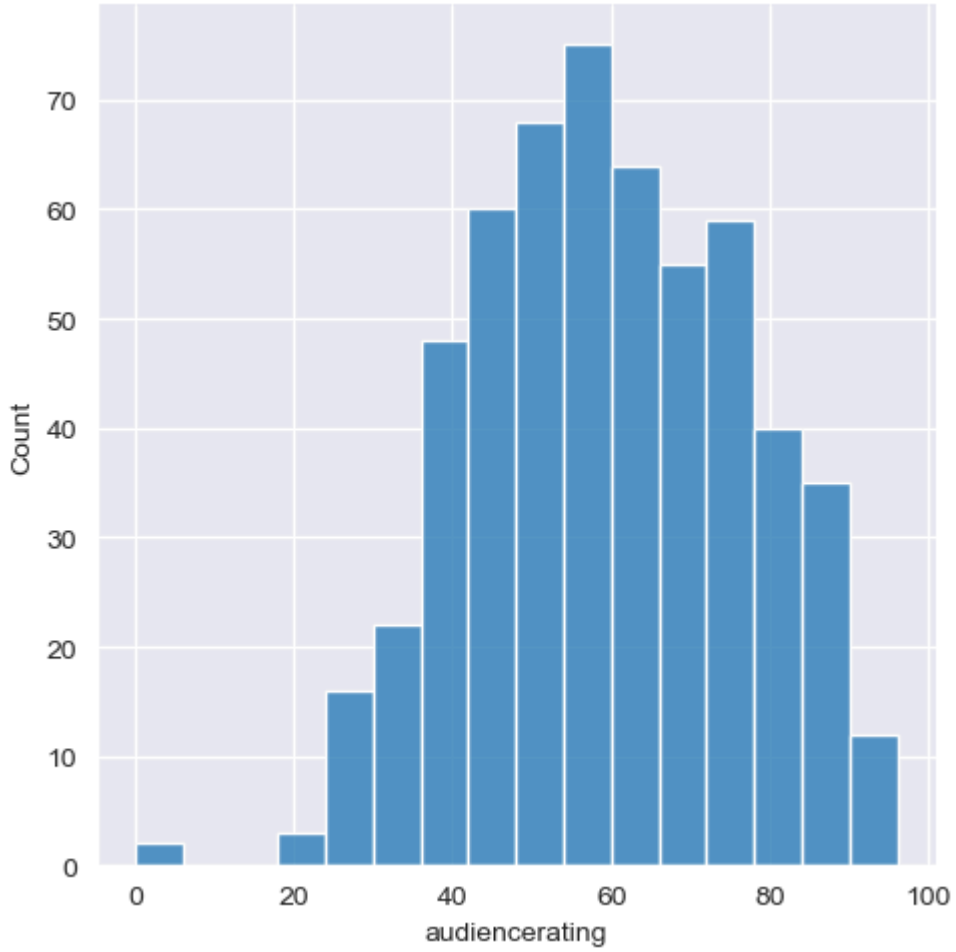
In [17]:
```
j=sns.jointplot(data=movie,x='rating',y='audiencerating')
plt.show()
```



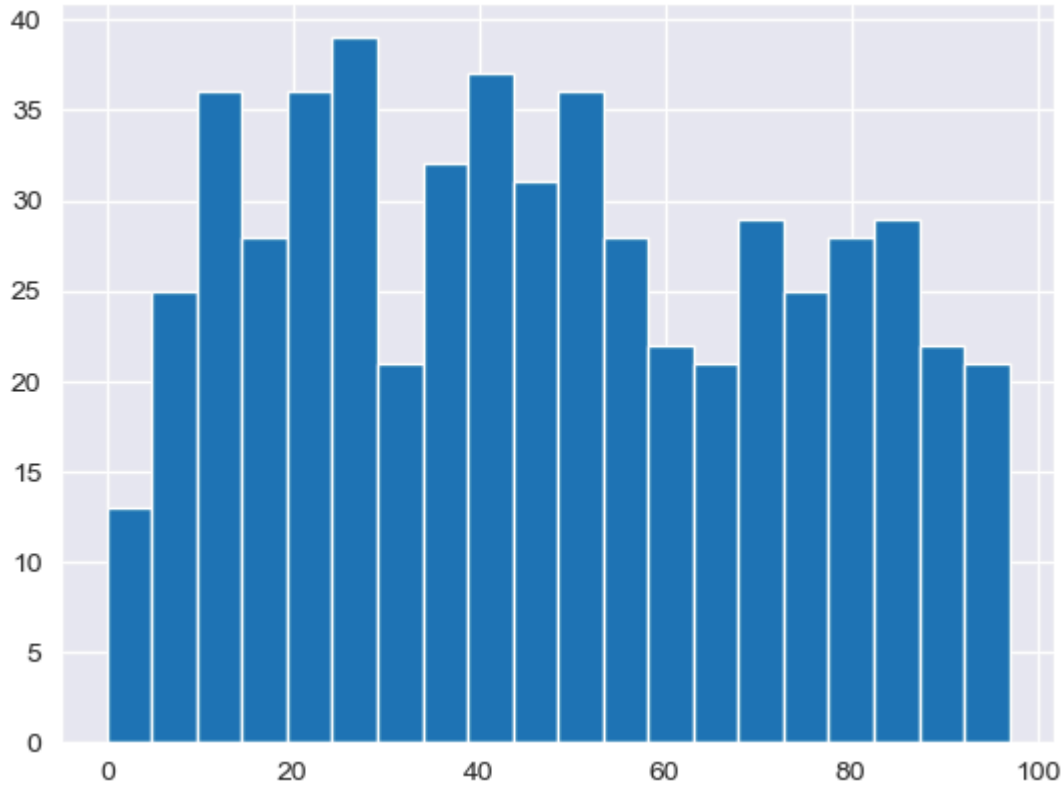
In [18]:
```
j=sns.jointplot(data=movie,x='rating',y='audiencerating',kind='hex')
sns.set_style('darkgrid')
plt.show()
```



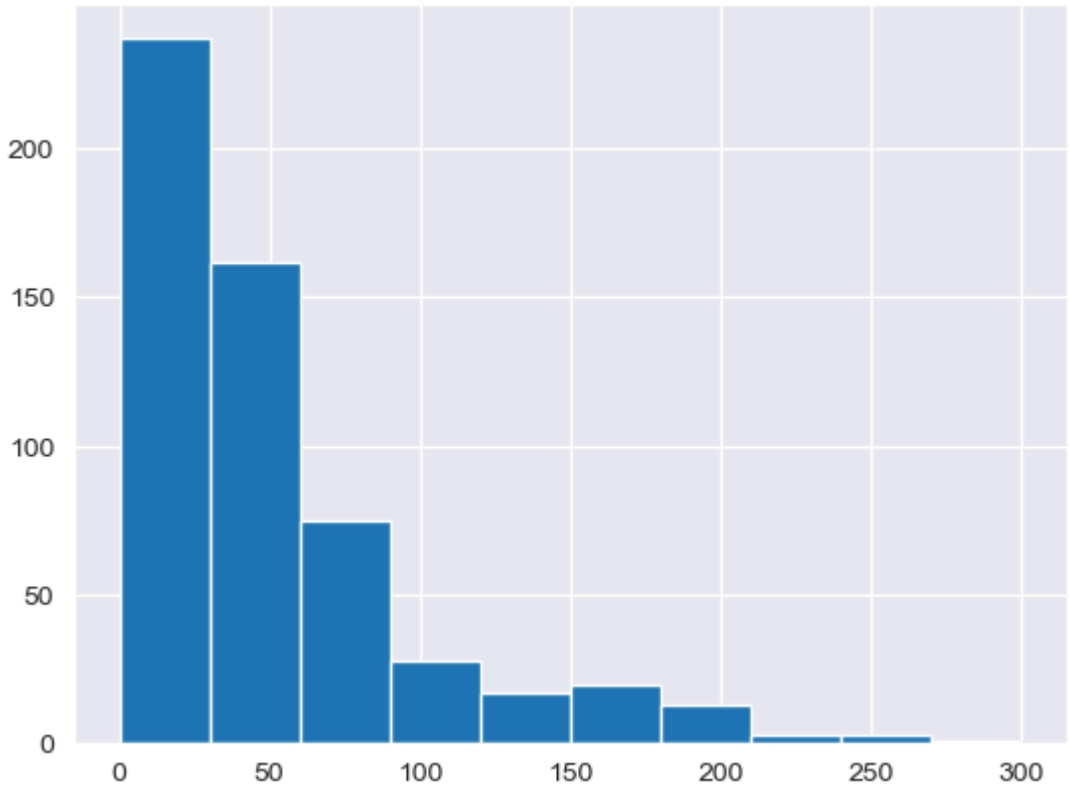
In [19]:
```
ml=sns.displot(movie.audiencerating)
sns.set_style('darkgrid')
plt.show()
```



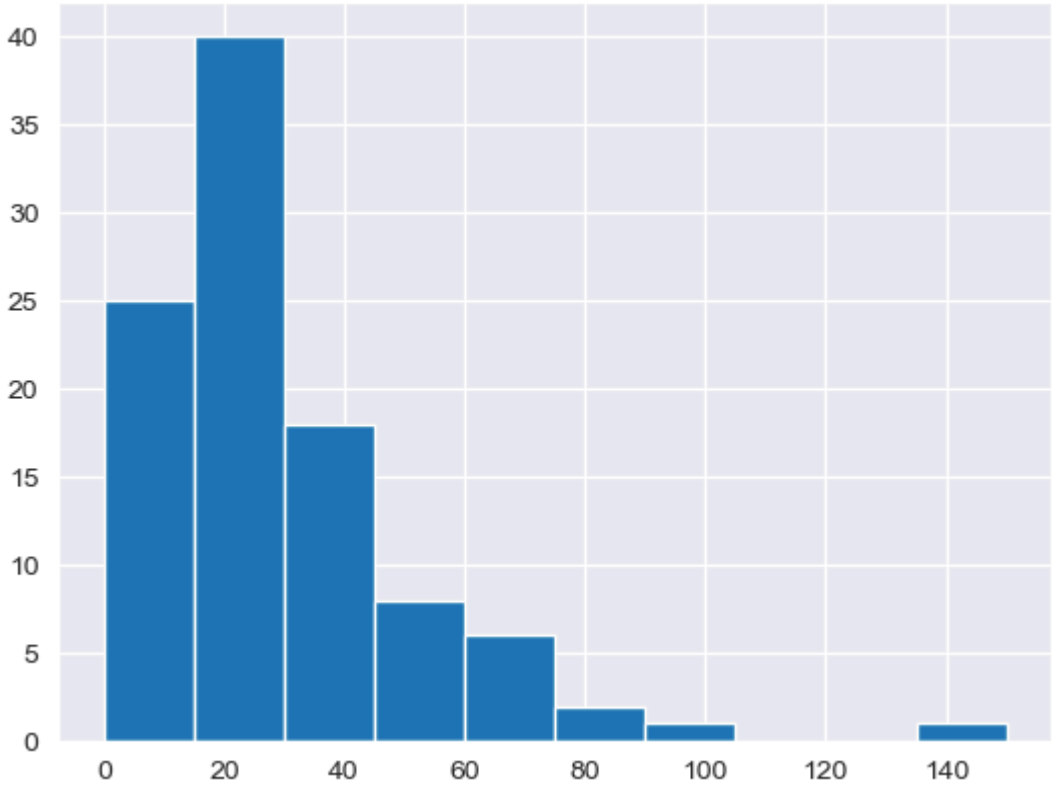
In [20]:
```
n1 = plt.hist(movie.rating, bins=20)
plt.show()
```



In [21]:
```
plt.hist(movie.budgetmillions)
plt.show()
```

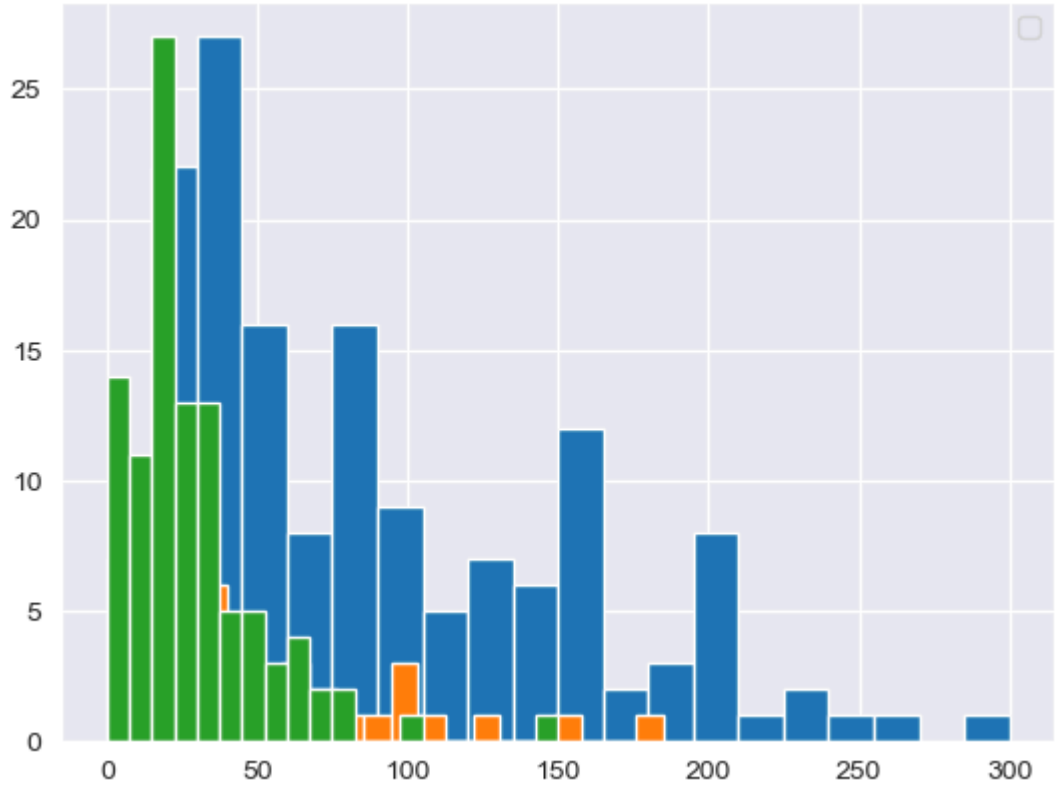


In [22]:
```
plt.hist(movie[movie.gener == 'Drama'].budgetmillions)
plt.show()
```

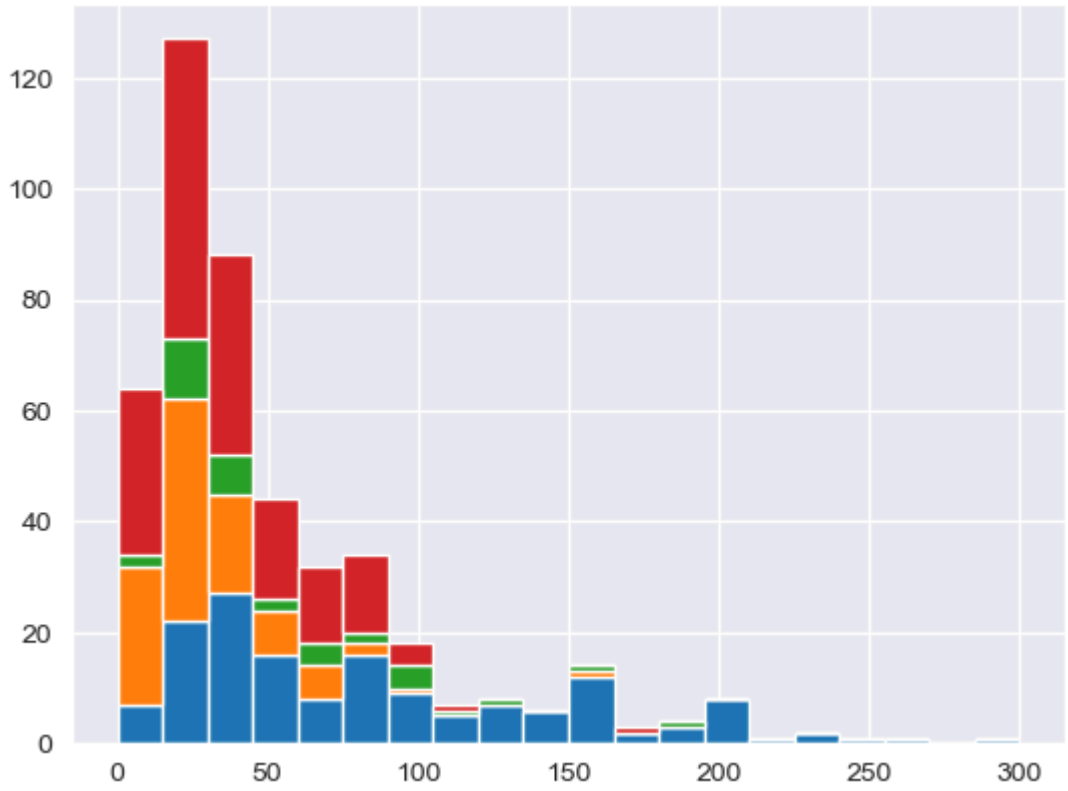


In [23]:
```
plt.hist(movie[movie.gener == 'Action'].budgetmillions, bins = 20)
plt.hist(movie[movie.gener == 'Thriller'].budgetmillions, bins = 20)
plt.hist(movie[movie.gener == 'Drama'].budgetmillions, bins = 20)
plt.legend()
plt.show()
```

```
C:\Users\Dell\AppData\Local\Temp\ipykernel_5084\3151234701.py:4: UserWarning: No
artists with labels found to put in legend.  Note that artists whose label start
with an underscore are ignored when legend() is called with no argument.
  plt.legend()
```



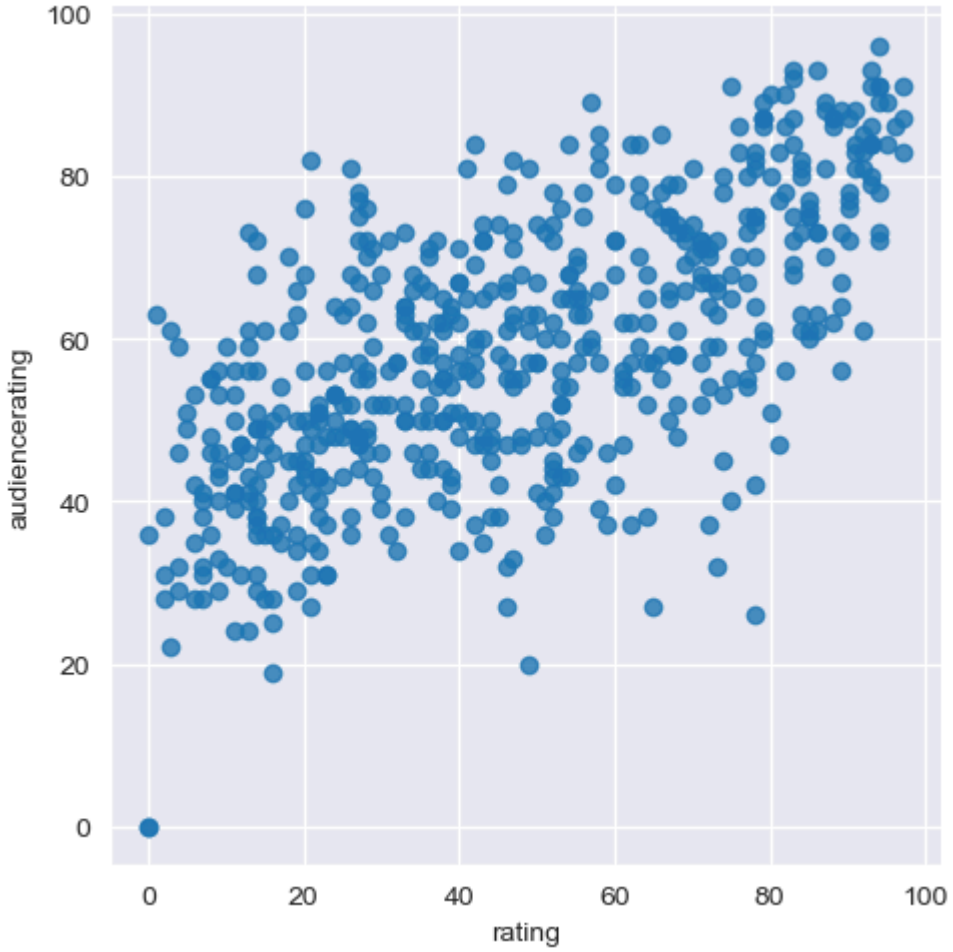
In [24]:
```
plt.hist([movie[movie.gener == 'Action'].budgetmillions,\
          movie[movie.gener == 'Drama'].budgetmillions, \
          movie[movie.gener == 'Thriller'].budgetmillions, \
          movie[movie.gener == 'Comedy'].budgetmillions],
         bins = 20, stacked = True)
plt.show()
```



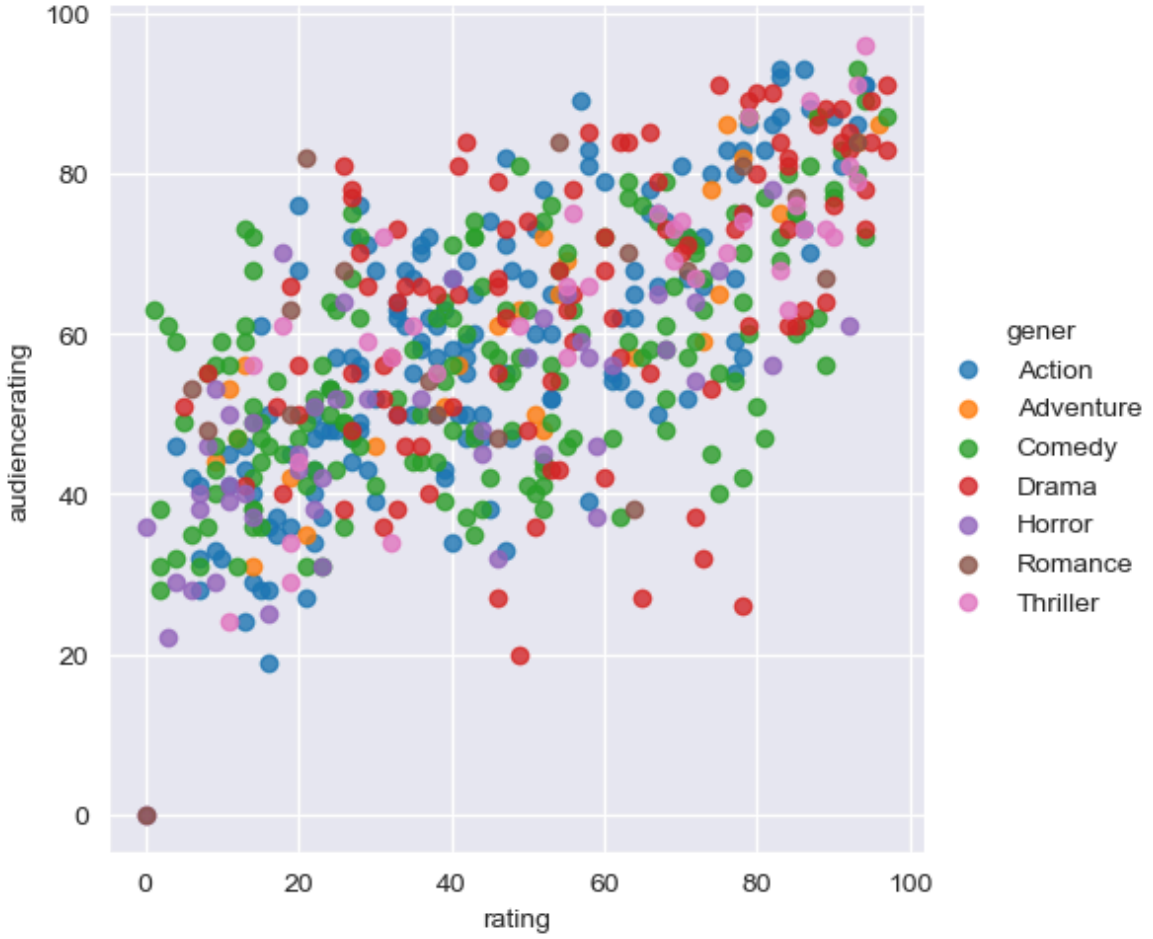
In [25]:
```
for gen in movie.gener.cat.categories:
    print(gen)
```

```
Action
Adventure
Comedy
Drama
Horror
Romance
Thriller
```

In [26]:
```
vis1 = sns.lmplot(data=movie, x='rating', y='audiencerating',\
                  fit_reg=False)
plt.show()
```



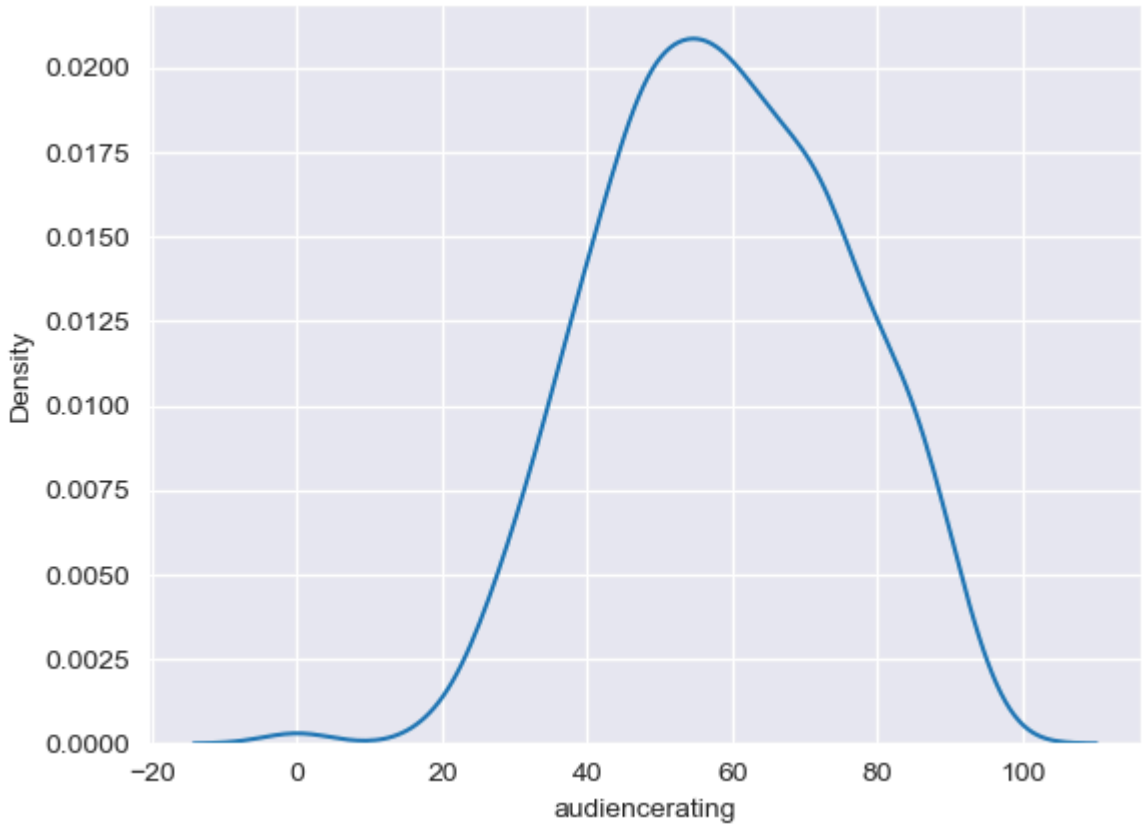
In [27]:
```
vis1 = sns.lmplot(data=movie, x='rating', y='audiencerating',\
                  fit_reg=False, hue = 'gener')
plt.show()
```



In [29]:
```
vis1 = sns.lmplot(data=movie, x='rating', y='audiencerating',\
                  fit_reg=False, hue = 'gener', size =10,aspect=1)
```

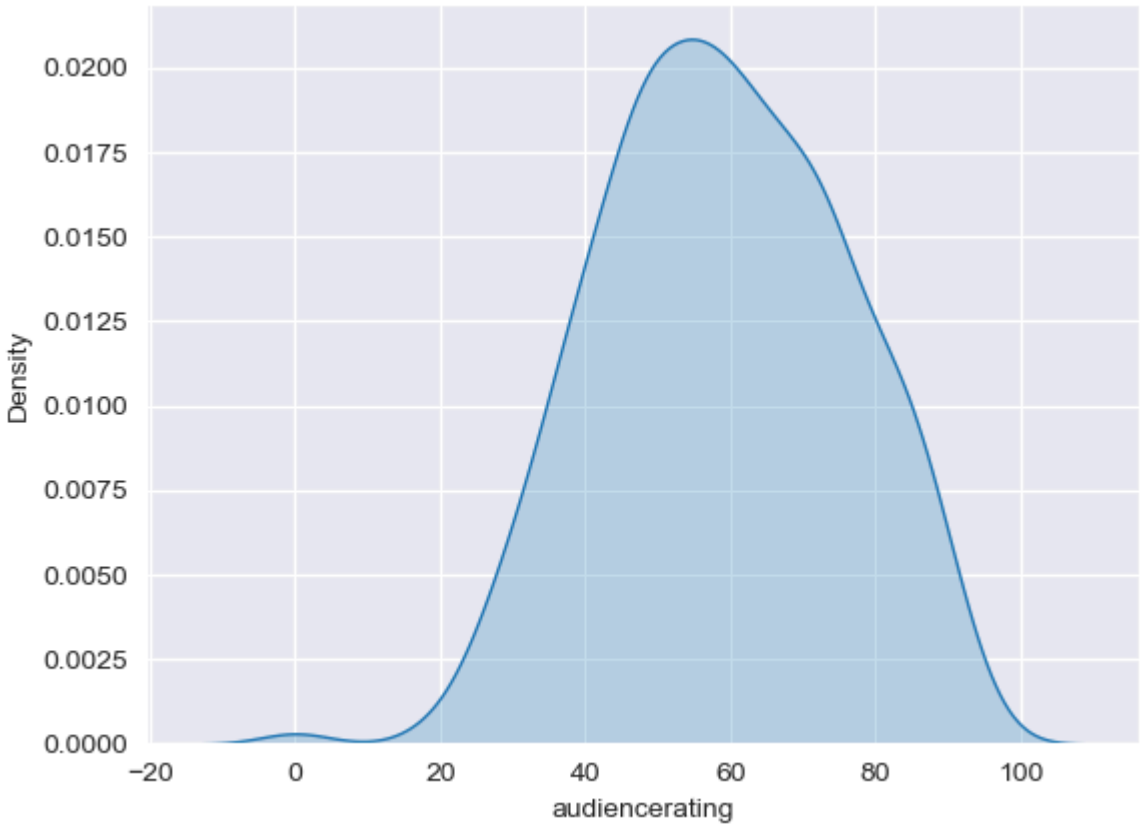
```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
Cell In[29], line 1
----> 1 vis1 = sns.lmplot(data=movie, x='rating', y='audiencerating',\
      2                   fit_reg=False, hue = 'gener', size =10,aspect=1)

TypeError: lmplot() got an unexpected keyword argument 'size'
```

In [31]:
```
k1 = sns.kdeplot(movie.audiencerating)
plt.show()
```

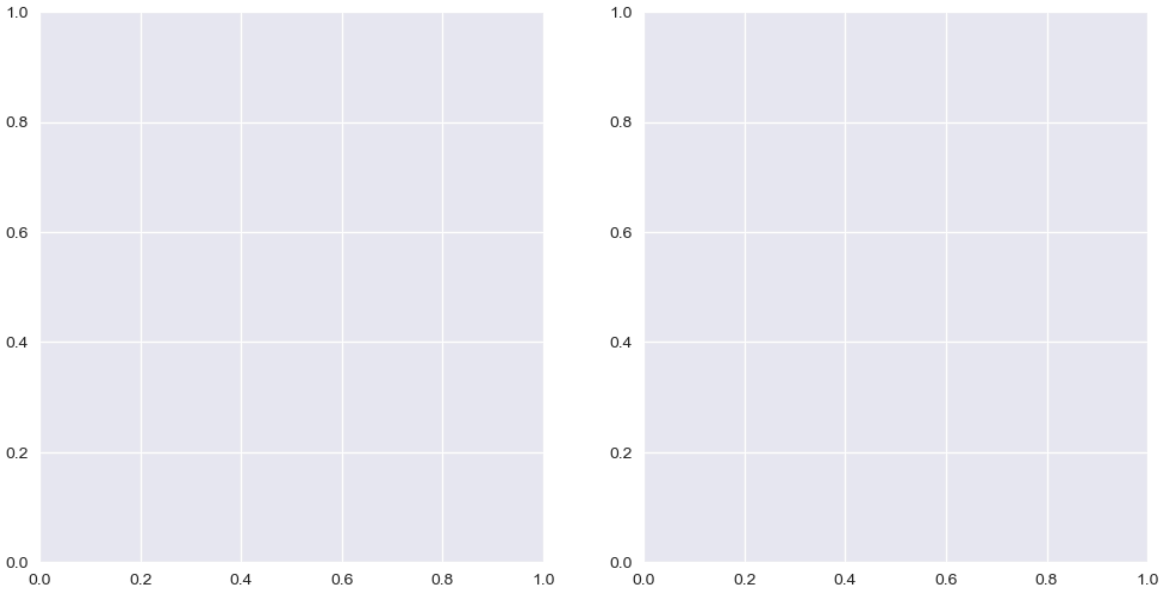


In [33]:
```
k1 = sns.kdeplot(movie.audiencerating,shade = True,shade_lowest=False,cmap='Reds
plt.show()
```

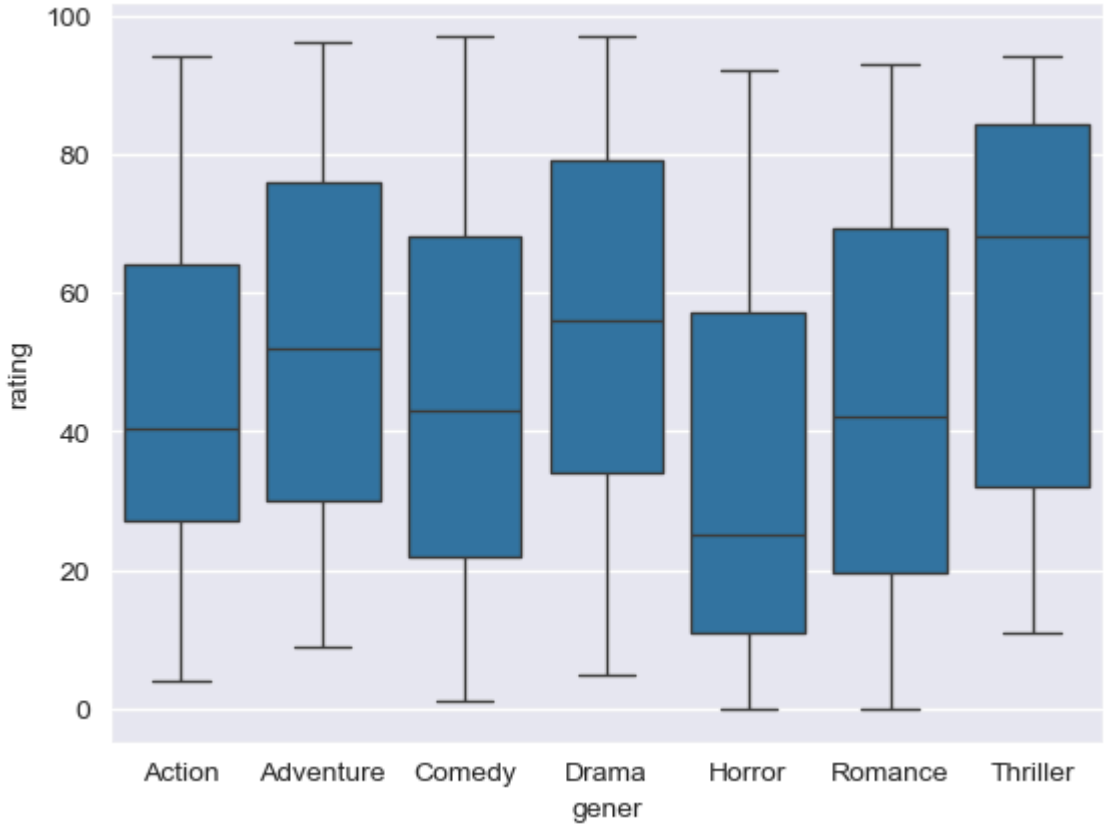
```
C:\Users\Dell\AppData\Local\Temp\ipykernel_5084\4067394718.py:1: UserWarning:

`shade_lowest` has been replaced by `thresh`; setting `thresh=0.05.
This will become an error in seaborn v0.14.0; please update your code.

  k1 = sns.kdeplot(movie.audiencerating,shade = True,shade_lowest=False,cmap='Red
s')
C:\Users\Dell\AppData\Local\Temp\ipykernel_5084\4067394718.py:1: FutureWarning:

`shade` is now deprecated in favor of `fill`; setting `fill=True`.
This will become an error in seaborn v0.14.0; please update your code.

  k1 = sns.kdeplot(movie.audiencerating,shade = True,shade_lowest=False,cmap='Red
s')
```



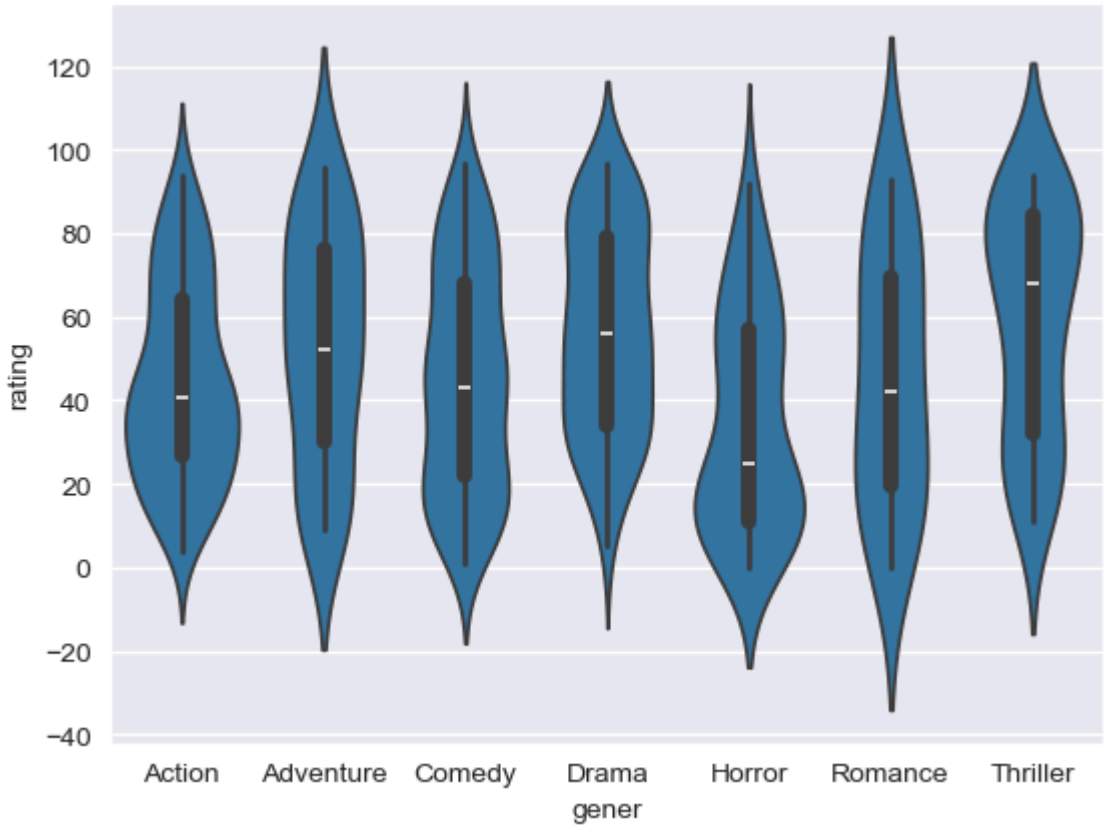
In [35]:
```
f, ax = plt.subplots(1,2, figsize =(12,6))
plt.show()
```



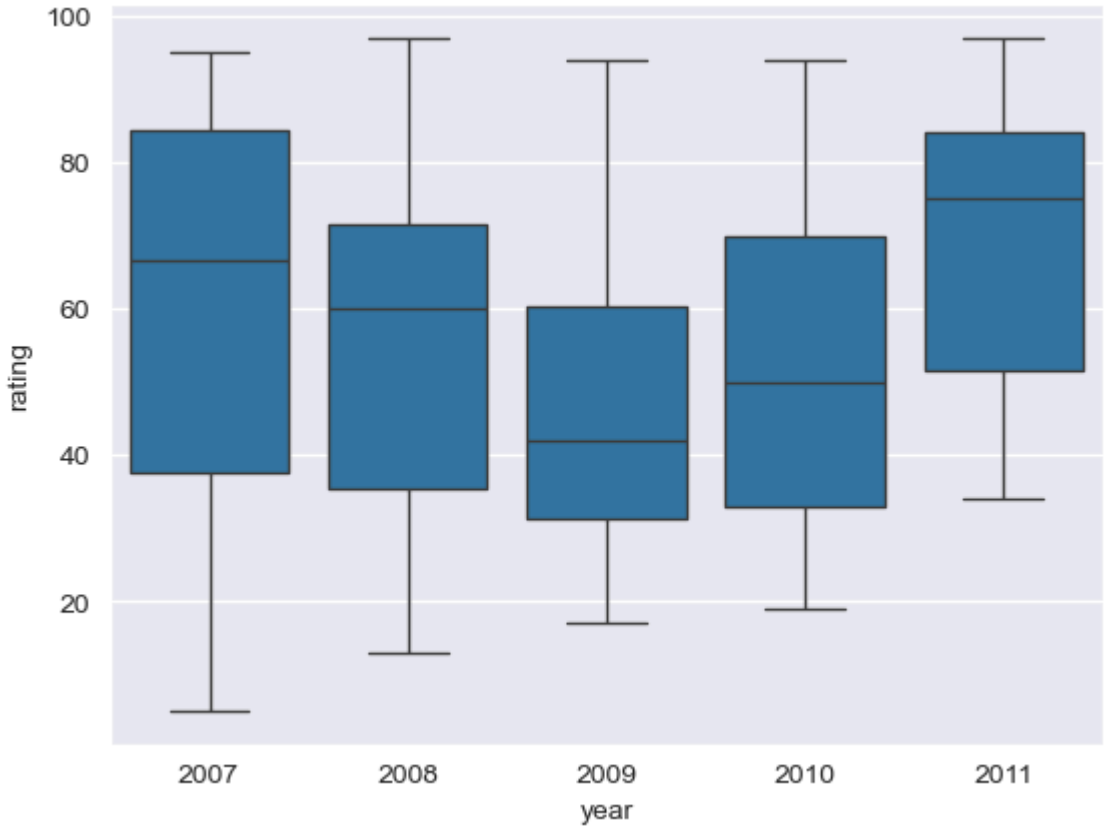
In [37]:
```
w = sns.boxplot(data=movie, x='gener', y = 'rating')
plt.show()
```



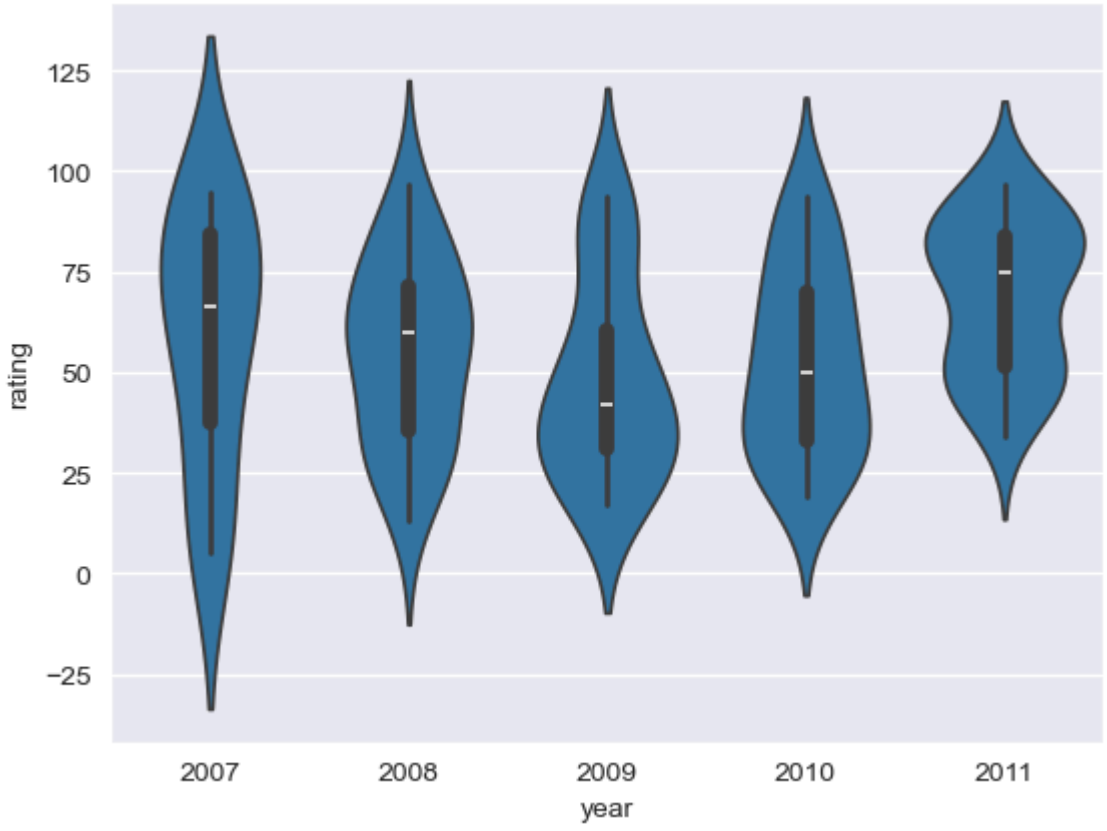
In [39]:
```
#violin plot

z = sns.violinplot(data=movie, x='gener', y = 'rating')
plt.show()
```



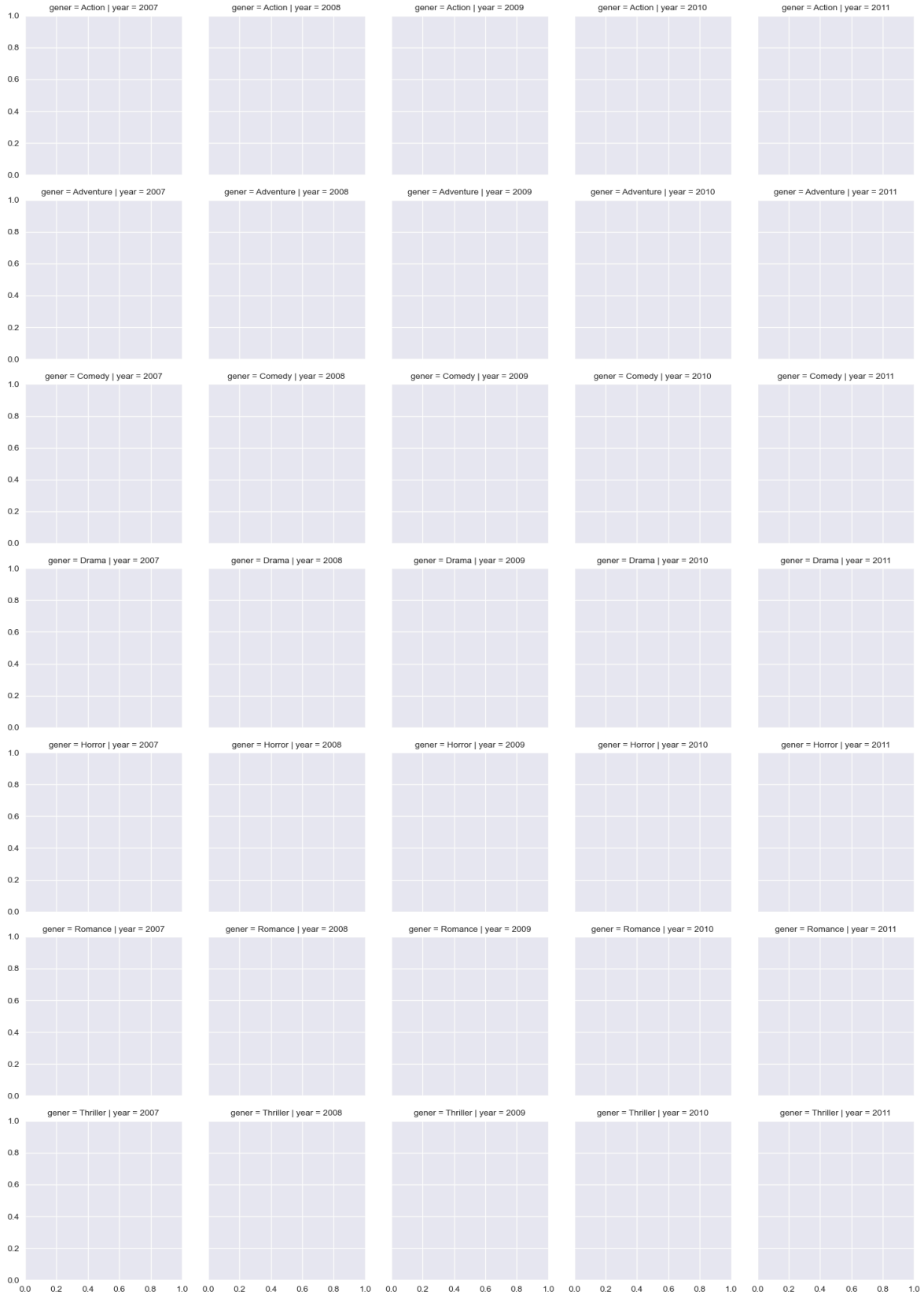
In [41]:
```
w1 = sns.boxplot(data=movie[movie.gener == 'Drama'], x='year', y = 'rating')
plt.show()
```



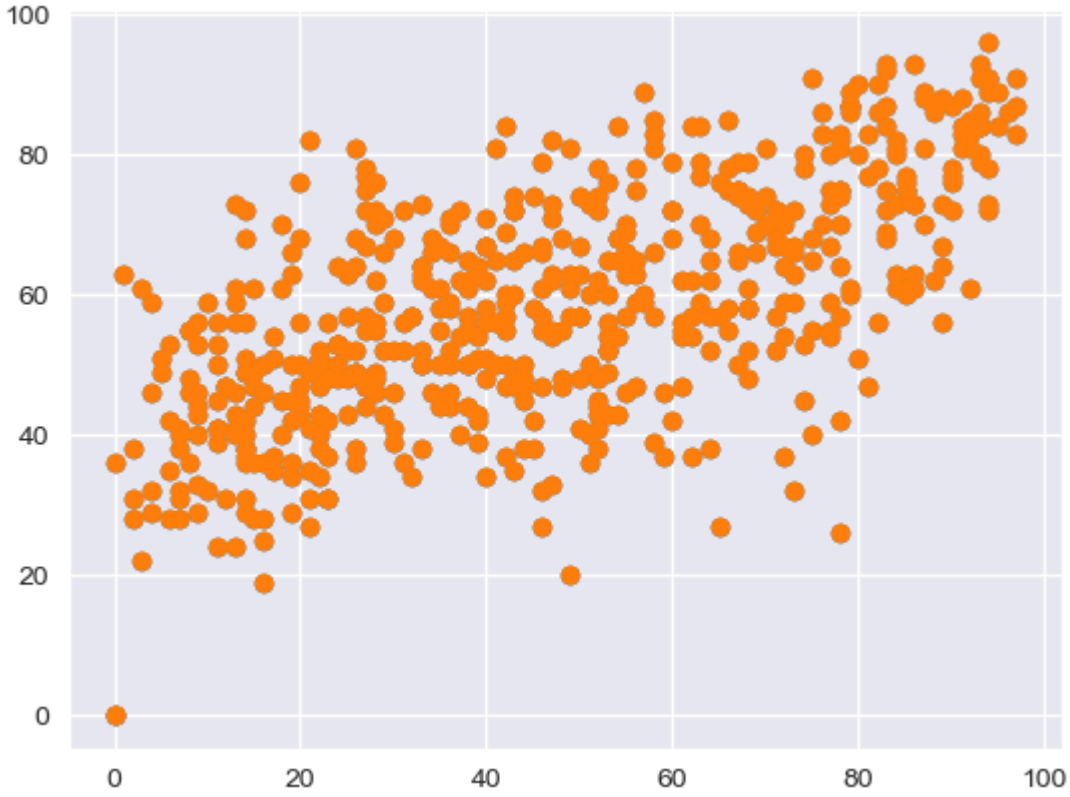
In [43]:
```
z = sns.violinplot(data=movie[movie.gener == 'Drama'], x='year', y = 'rating')
plt.show()
```



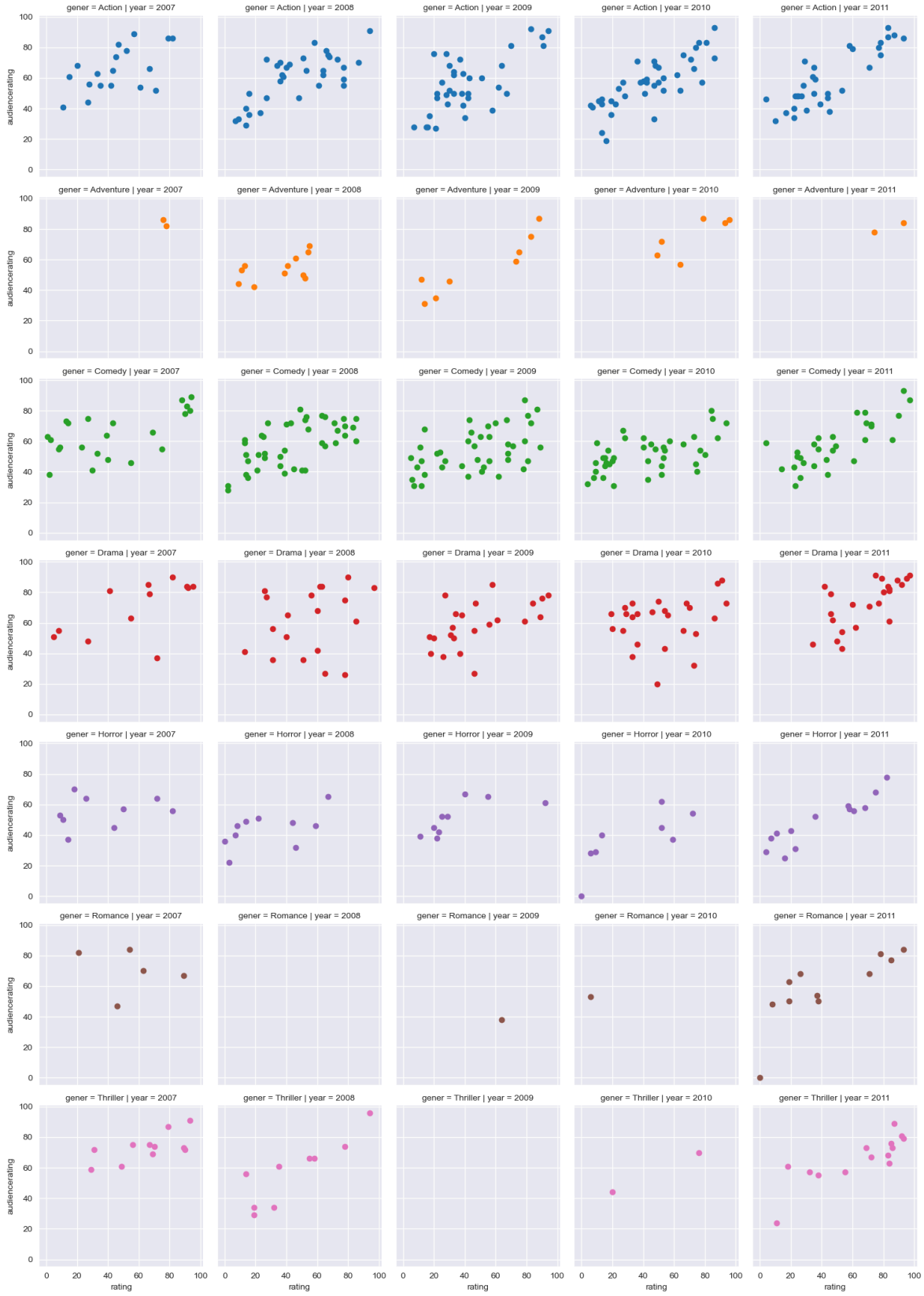
In [45]:
```
g =sns.FacetGrid (movie, row = 'gener', col = 'year', hue = 'gener') #kind of su
plt.show()
```

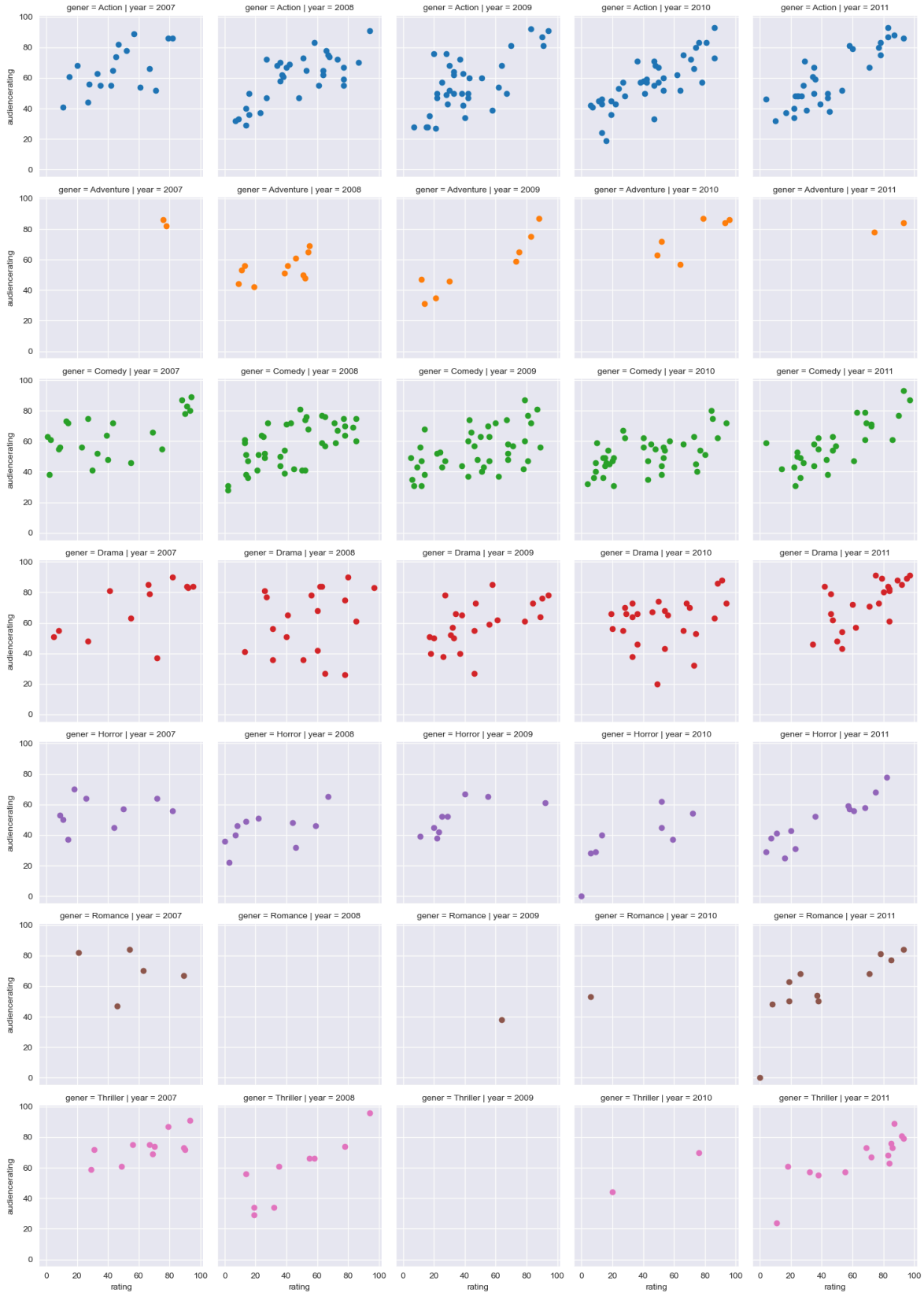


In [50]:
```
plt.scatter(movie.rating,movie.audiencerating)
plt.show()
```

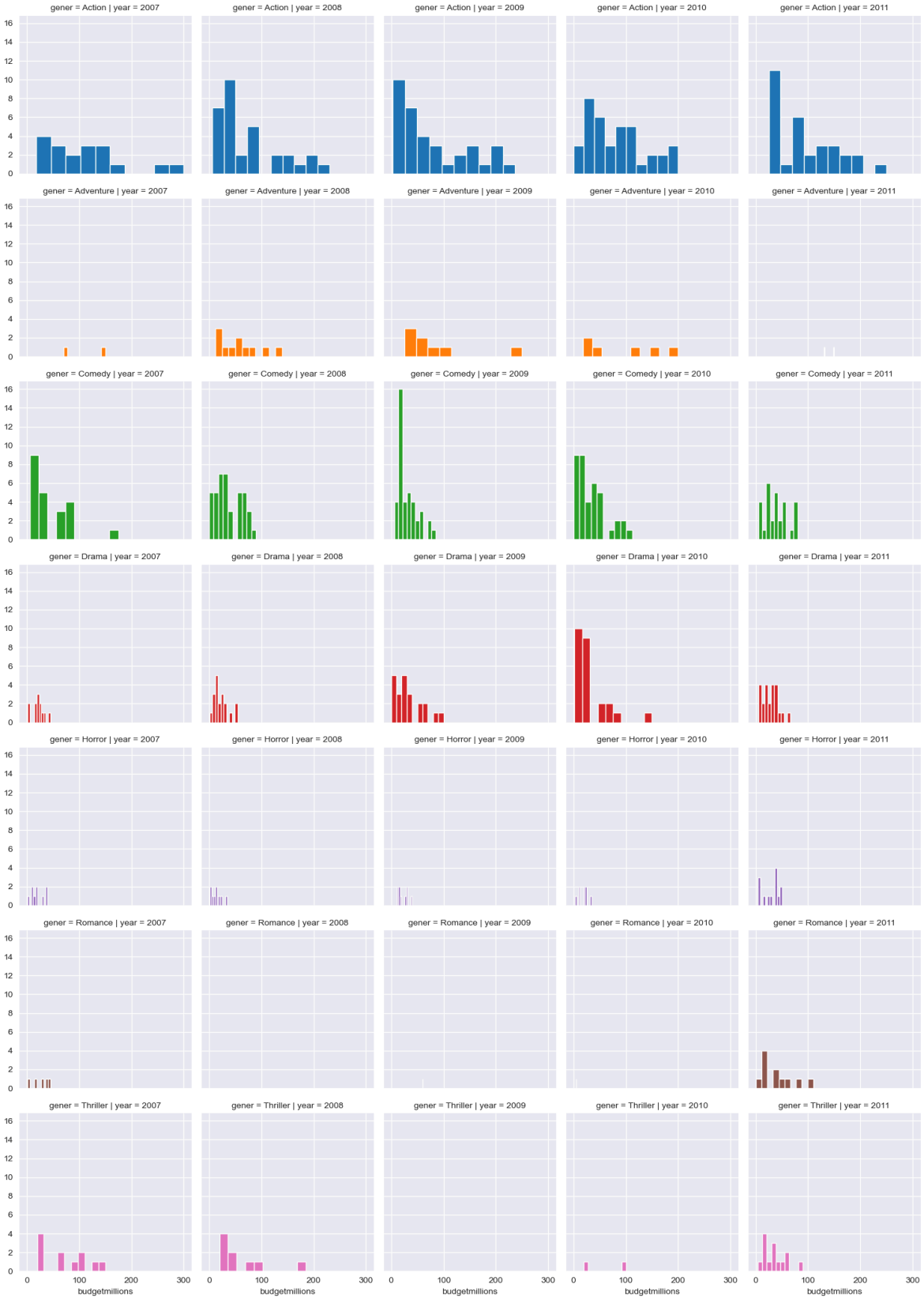


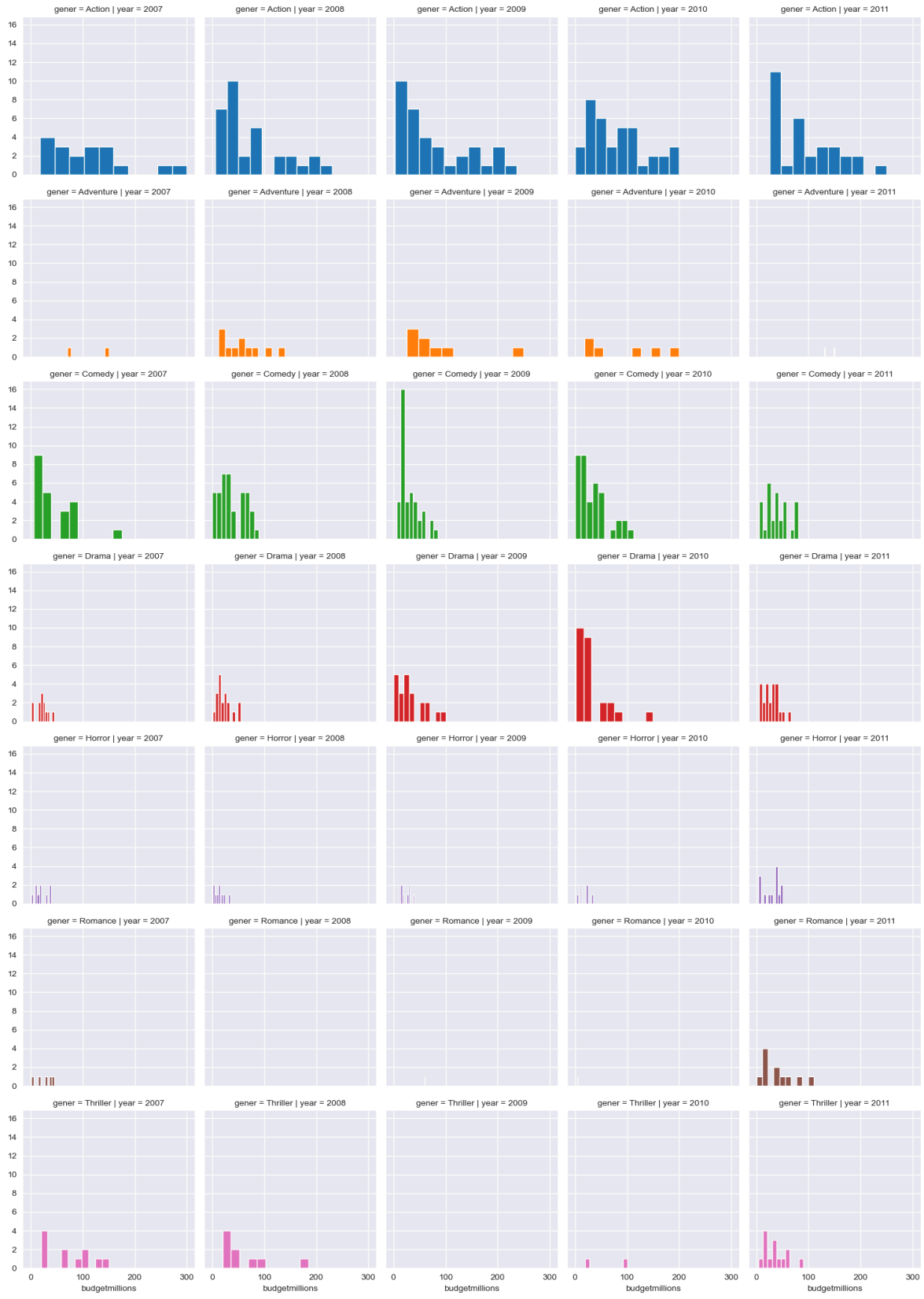
In [54]:
```
g =sns.FacetGrid (movie, row = 'gener', col = 'year', hue = 'gener')
g = g.map(plt.scatter, 'rating', 'audiencerating' ) #scatterplots are mapped in
plt.show()
```


gener = Action | year = 2007
gener = Action | year = 2008
gener = Action | year = 2009
gener = Action | year = 2010
gener = Action | year = 2011
gener = Adventure | year = 2007
gener = Adventure | year = 2008
gener = Adventure | year = 2009
gener = Adventure | year = 2010
gener = Adventure | year = 2011
gener = Comedy | year = 2007
gener = Comedy | year = 2008
gener = Comedy | year = 2009
gener = Comedy | year = 2010
gener = Comedy | year = 2011
gener = Drama | year = 2007
gener = Drama | year = 2008
gener = Drama | year = 2009
gener = Drama | year = 2010
gener = Drama | year = 2011
gener = Horror | year = 2007
gener = Horror | year = 2008
gener = Horror | year = 2009
gener = Horror | year = 2010
gener = Horror | year = 2011
gener = Romance | year = 2007
gener = Romance | year = 2008
gener = Romance | year = 2009
gener = Romance | year = 2010
gener = Romance | year = 2011
gener = Thriller | year = 2007
gener = Thriller | year = 2008
gener = Thriller | year = 2009
gener = Thriller | year = 2010
gener = Thriller | year = 2011
audiencerating
rating
0
20
40
60
80
100

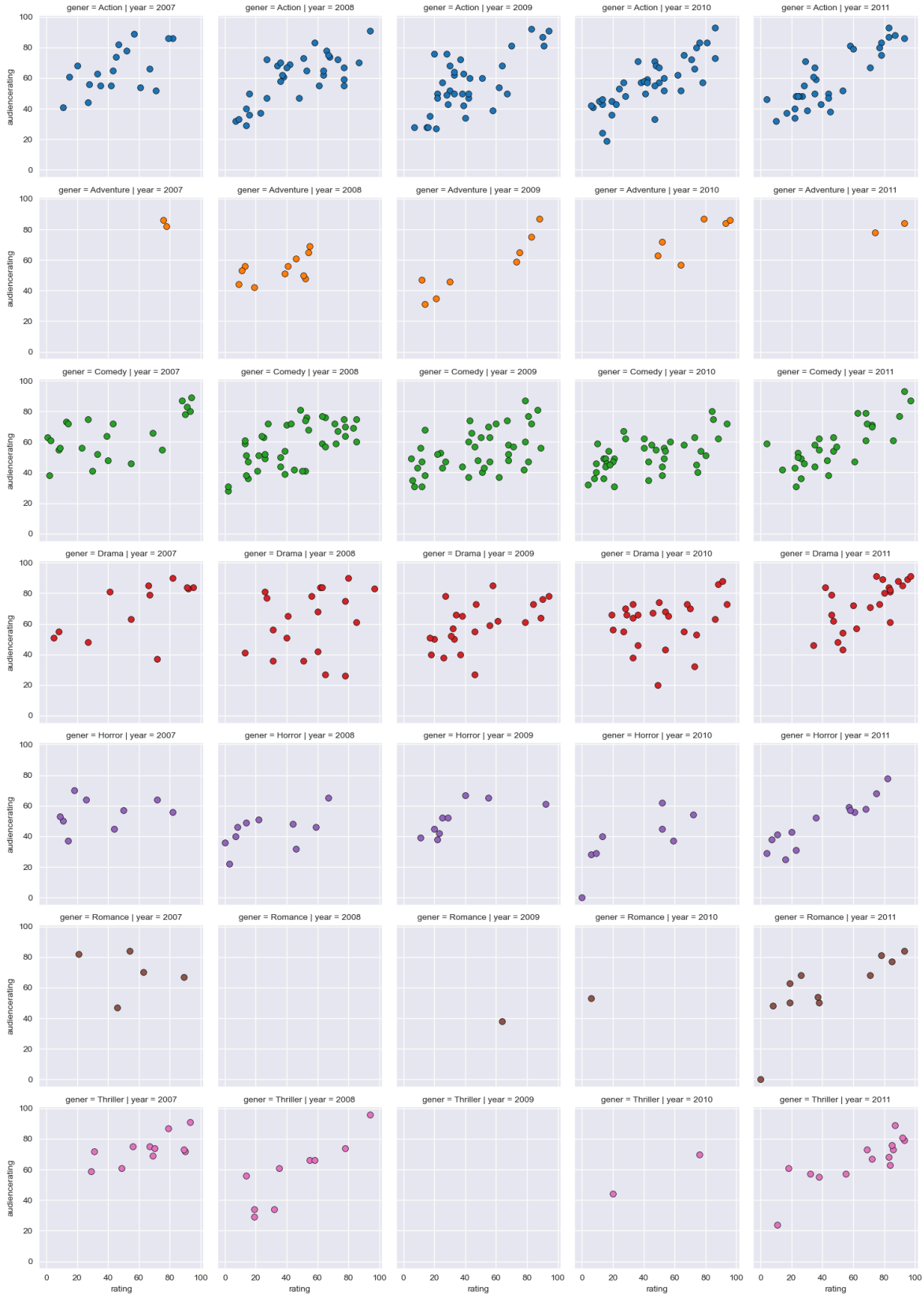


In [58]:
```
# you can populated any type of chat.

g =sns.FacetGrid (movie, row = 'gener', col = 'year', hue = 'gener')
g = g.map(plt.hist, 'budgetmillions') #scatterplots are mapped in facetgrid
plt.show()
```


gener = Action | year = 2007
gener = Action | year = 2008
gener = Action | year = 2009
gener = Action | year = 2010
gener = Action | year = 2011
gener = Adventure | year = 2007
gener = Adventure | year = 2008
gener = Adventure | year = 2009
gener = Adventure | year = 2010
gener = Adventure | year = 2011
gener = Comedy | year = 2007
gener = Comedy | year = 2008
gener = Comedy | year = 2009
gener = Comedy | year = 2010
gener = Comedy | year = 2011
gener = Drama | year = 2007
gener = Drama | year = 2008
gener = Drama | year = 2009
gener = Drama | year = 2010
gener = Drama | year = 2011
gener = Horror | year = 2007
gener = Horror | year = 2008
gener = Horror | year = 2009
gener = Horror | year = 2010
gener = Horror | year = 2011
gener = Romance | year = 2007
gener = Romance | year = 2008
gener = Romance | year = 2009
gener = Romance | year = 2010
gener = Romance | year = 2011
gener = Thriller | year = 2007
gener = Thriller | year = 2008
gener = Thriller | year = 2009
gener = Thriller | year = 2010
gener = Thriller | year = 2011
budgetmillions



In [60]:
```
g =sns.FacetGrid (movie, row = 'gener', col = 'year', hue = 'gener')
kws = dict(s=50, linewidth=0.5,edgecolor='black')
g = g.map(plt.scatter, 'rating', 'audiencerating',**kws ) #scatterplots are mapp
plt.show()
```



In [62]:
```
sns.set_style('darkgrid')
f, axes = plt.subplots (2,2, figsize = (15,15))

k1 = sns.kdeplot(movie.budgetmillions,movie.audiencerating,ax=axes[0,0])
k2 = sns.kdeplot(movie.budgetmillions,movie.rating,ax = axes[0,1])

k1.set(xlim=(-20,160))
k2.set(xlim=(-20,160))

z = sns.violinplot(data=movie[movie.gener=='Drama'], x='year', y = 'rating', ax=
```

```
k4 = sns.kdeplot(movie.rating,movie.audiencerating,shade = True,shade_lowest=Fal

k4b = sns.kdeplot(movie.rating, movies.audiencerating,cmap='Reds',ax = axes[1,1]

plt.show()
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
Cell In[62], line 4
      1 sns.set_style('darkgrid')
      2 f, axes = plt.subplots (2,2, figsize = (15,15))
----> 4 k1 = sns.kdeplot(movie.budgetmillions,movie.audiencerating,ax=axes[0,0])
      5 k2 = sns.kdeplot(movie.budgetmillions,movie.rating,ax = axes[0,1])
      7 k1.set(xlim=(-20,160))

TypeError: kdeplot() takes from 0 to 1 positional arguments but 2 positional argu
ments (and 1 keyword-only argument) were given
```

In [ ]: